1. **Clases locales y globales**

Las clases globales se definen en el generador de clases, SE24. Estas son almacenadas en el Class Pools en la librería de clases en el Repositorio R/3. Las clases locales se definen en un programa ABAP y solo pueden ser invocadas desde el programa en el que se han definido.

Si se define una clase que se va a usar en un único programa, normalmente es suficiente con definir los componentes visibles de manera que la clase se ajuste a nuestro programa. Por orto lado, las clases globales deben estar preparadas para ser usadas en cualquier parte.

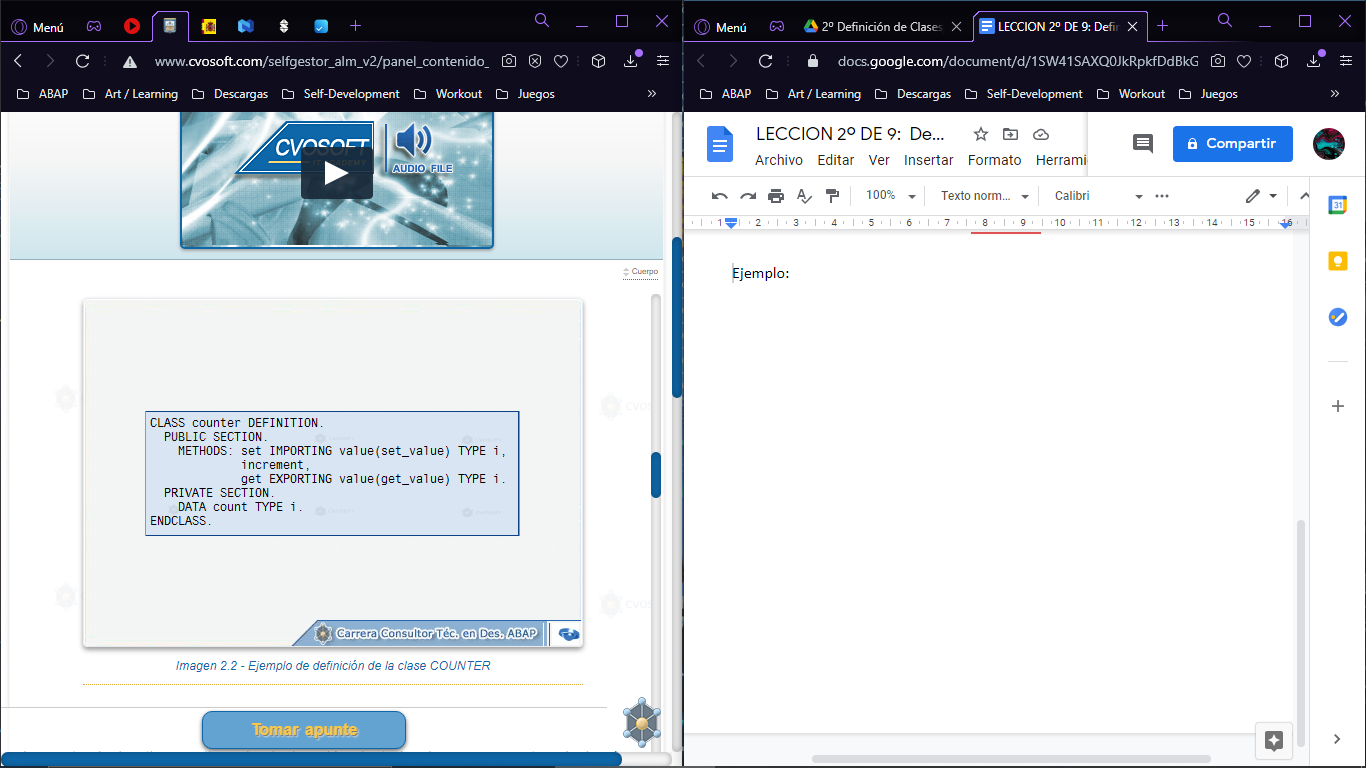
1. Definición de Clases locales

Las clases locales son el conjunto de sentencias que están entre las sentencias **CLASS...ENDCLASS**. Consta de dos clases de una parte declarativa, en la que se definen los componentes, y una parte de la implementación. La parte declarativa está comprendida entre las sentencias:

# CLASS <class> DEFINITION.

# …

# ENDCLASS.

Ejemplo: 

La parte declarativa contiene la declaración de todos los componentes de la clase (atributos, métodos y eventos). Cuando se definen clases locales, la parte declarativa pertenece a los datos globales, por tanto se habrá de situar al principio del programa

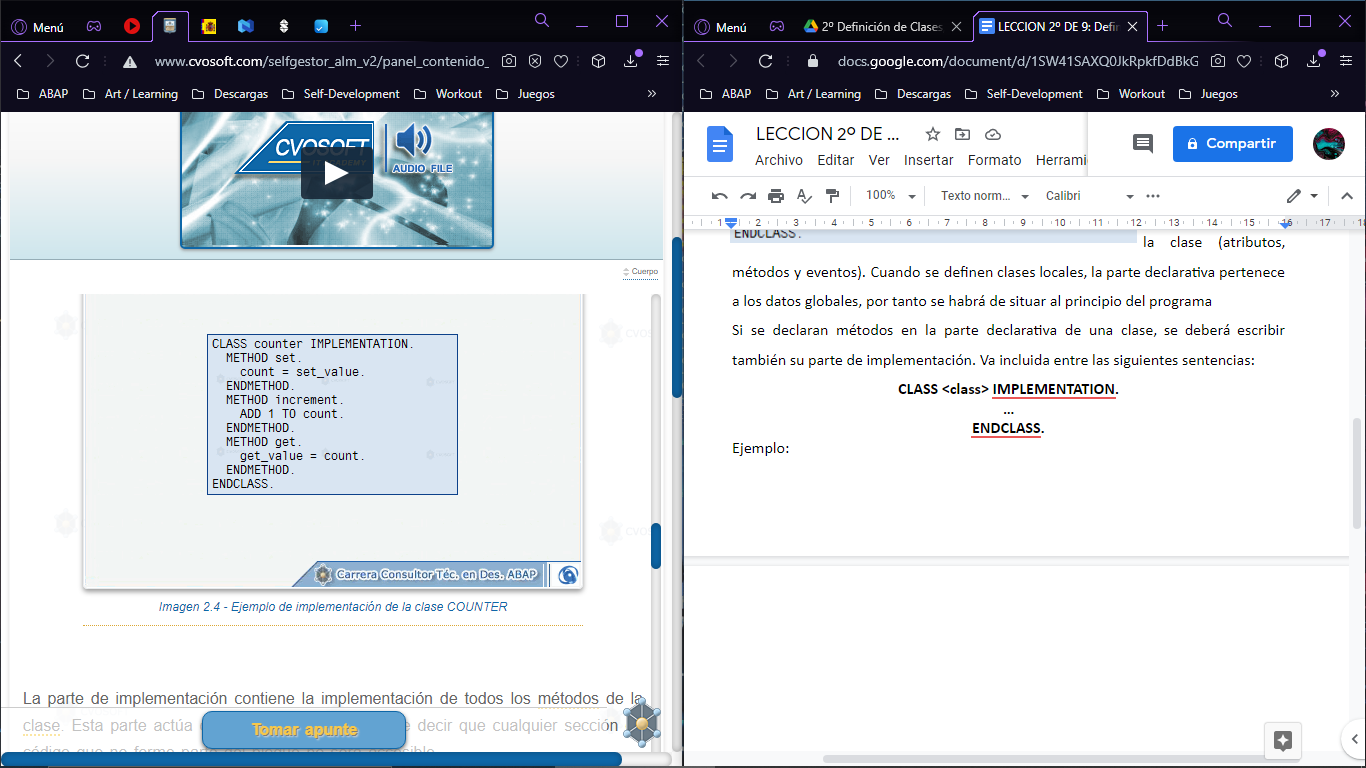
Si se declaran métodos en la parte declarativa de una clase, se deberá escribir también su parte de implementación. Va incluida entre las siguientes sentencias:

# **CLASS <class> IMPLEMENTATION.**

# **…**

# **ENDCLASS.**

Ejemplo:



Esta parte actúa como un bloque, esto quiere decir que cualquier sección de código que no forme parte del bloque no será accesible.

1. **Estructura y Componentes de una Clase**

Una clase se define principalmente basándose en:

* Una clase contiene componentes.
* Cada componente se asigna a una sección de visibilidad (público, protegido o privado).
* Las clases implementan métodos.

Todos los componentes son declarados en la parte declarativa de la clase. Los componentes definen los atributos de los objetos en una clase.

Cada componente es asignado a una de las tres distintas secciones de visibilidad que definen la interface externa de la clase. (Interface es una estructura que se puede implementar en una clase para extender el ámbito de esa clase).

Todos los componentes de una clase son visibles dentro de la clase y todos comparten el mismo espacio por lo que sus nombres deben ser únicos dentro de la clase. Hay 2 tipos de componentes en una clase:

* **Dependientes de instancia:** existen separadamente para cada objeto de una clase
* **Independientes de instancia:** existen solo una vez para la clase entera

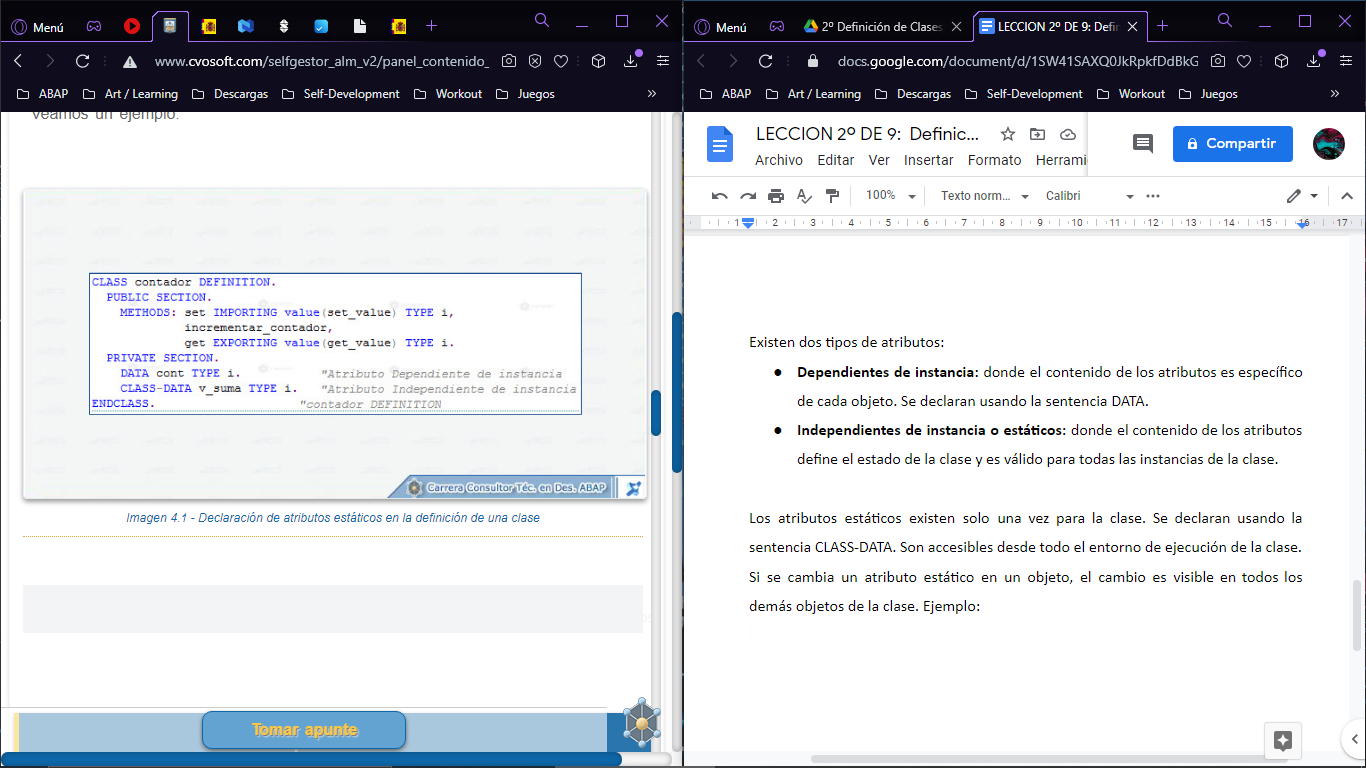
1. **Atributos**

Son los campos de datos internos de una clase y pueden tener cualquier tipo de datos ABAP. El estado de un objeto viene determinado por el contenido de sus atributos. Un tipo de atributos don las variables referenciales. Estas variables permiten crear y acceder a los objetos, de manera que si se definen en una clase permiten acceder a otros objetos desde dentro de la clase.

Existen dos tipos de atributos:

* **Dependientes de instancia:** donde el contenido de los atributos es específico de cada objeto. Se declaran usando la sentencia DATA.
* **Independientes de instancia o estáticos:** donde el contenido de los atributos define el estado de la clase y es válido para todas las instancias de la clase.

Los atributos estáticos existen solo una vez para la clase. Se declaran usando la sentencia CLASS-DATA. Son accesibles desde todo el entorno de ejecución de la clase. Si se cambia un atributo estático en un objeto, el cambio es visible en todos los demás objetos de la clase. Ejemplo:



1. **Métodos**

Son procedimientos internos de una clase que definen el comportamiento de un objeto. Los métodos pueden acceder a todos los atributos de una clase. Esto les permite cambiar el contenido de los atributos de un objeto.

Los métodos poseen también un interface con parámetros que les permite recibir valores cuando son invocados y devolver valores después de la llamada. Los atributos privados de una clase solo pueden ser cambiados por métodos de la misma clase.

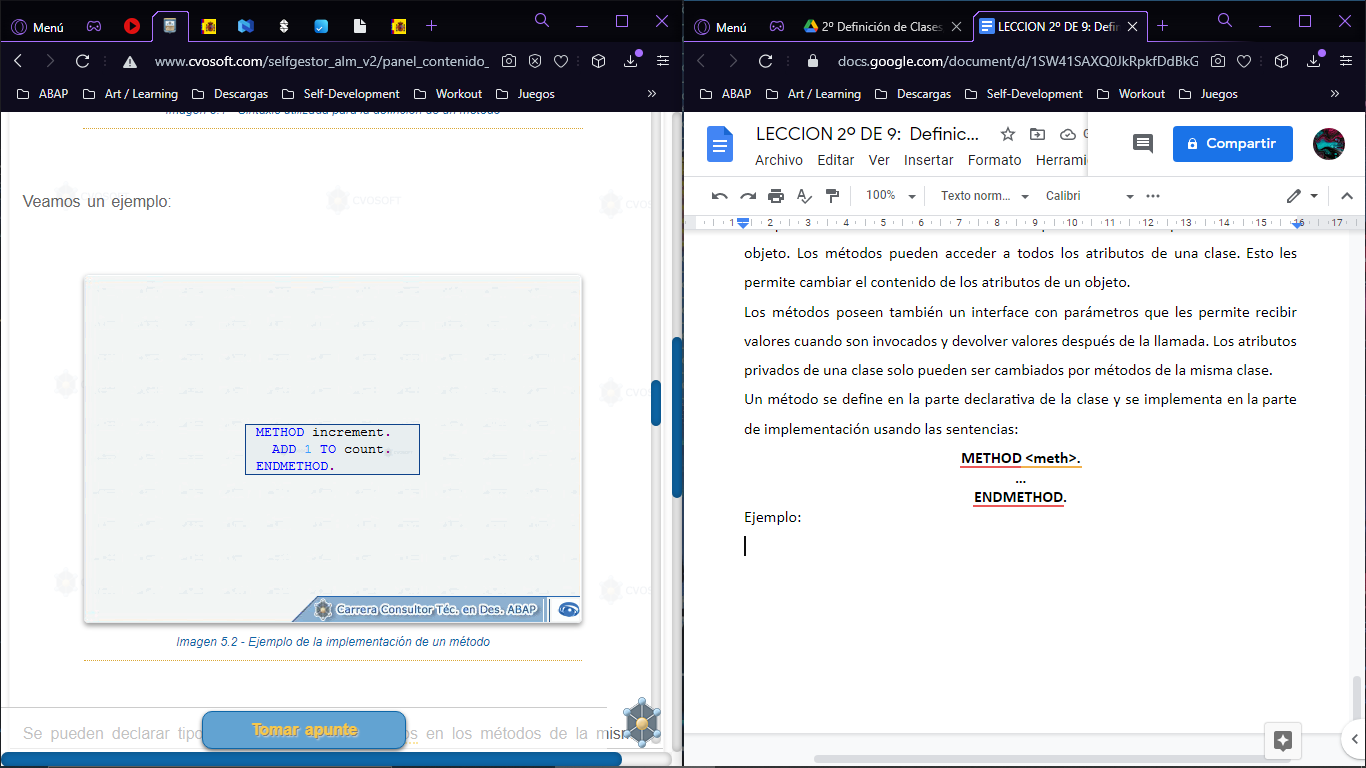
Un método se define en la parte declarativa de la clase y se implementa en la parte de implementación usando las sentencias:

# METHOD <meth>.

# …

# ENDMETHOD.

Ejemplo:



Se pueden declarar tipos de datos locales y objetos en los métodos de la misma manera que en cualquier otro procedimiento ABAP. Los métodos se llaman mediante le sentencia **CALL METHOD**.

Existen tres tipos de métodos:

* **Dependientes de instancia:** Estos métodos se declaran usando la sentencia **METHODS**. Pueden acceder a todos los atributos de una clase y pueden desencadenar todos los eventos de una clase.
* **Independientes de instancia o estáticos:** estos métodos se declaran usando la sentencia **CLASS-METHODS**. Solo pueden acceder a los atributos estáticos y desencadenar eventos estáticos.
* **Especiales:** además de los métodos normales que se pueden llamar con la sentencia **CALL METHOD**, hay dos métodos especiales llamados CONSTRUCTOR y **CLASS\_CONSTRUCTOR** que son automáticamente llamados cuando se crea un objeto (CONSTRUCTOR) o cuando se accede por primera vez a los componentes de la clase (CLASS\_CONSTRUCTOR).

1. **Eventos**

Son acontecimientos o sucesos que utilizan los objetos o las clases para desencadenar la ejecución de *un método en otro objeto* o clase. Estos métodos se llaman métodos que manejan eventos (event handler methods).

En una llamada normal a un método puede ser llamado por cualquier número de usuarios. Cuando un evento es desencadenado, cualquier número de estos métodos puede ser llamado.

La unión entre el disparador del evento **“trigger”** y el manejador del evento “handler” no es establecida de antemano, sino en el entorno de ejecución. En las llamadas normales a métodos, el programa que llama determina los métodos a los que quiere llamar. Estos métodos tienen que existir.

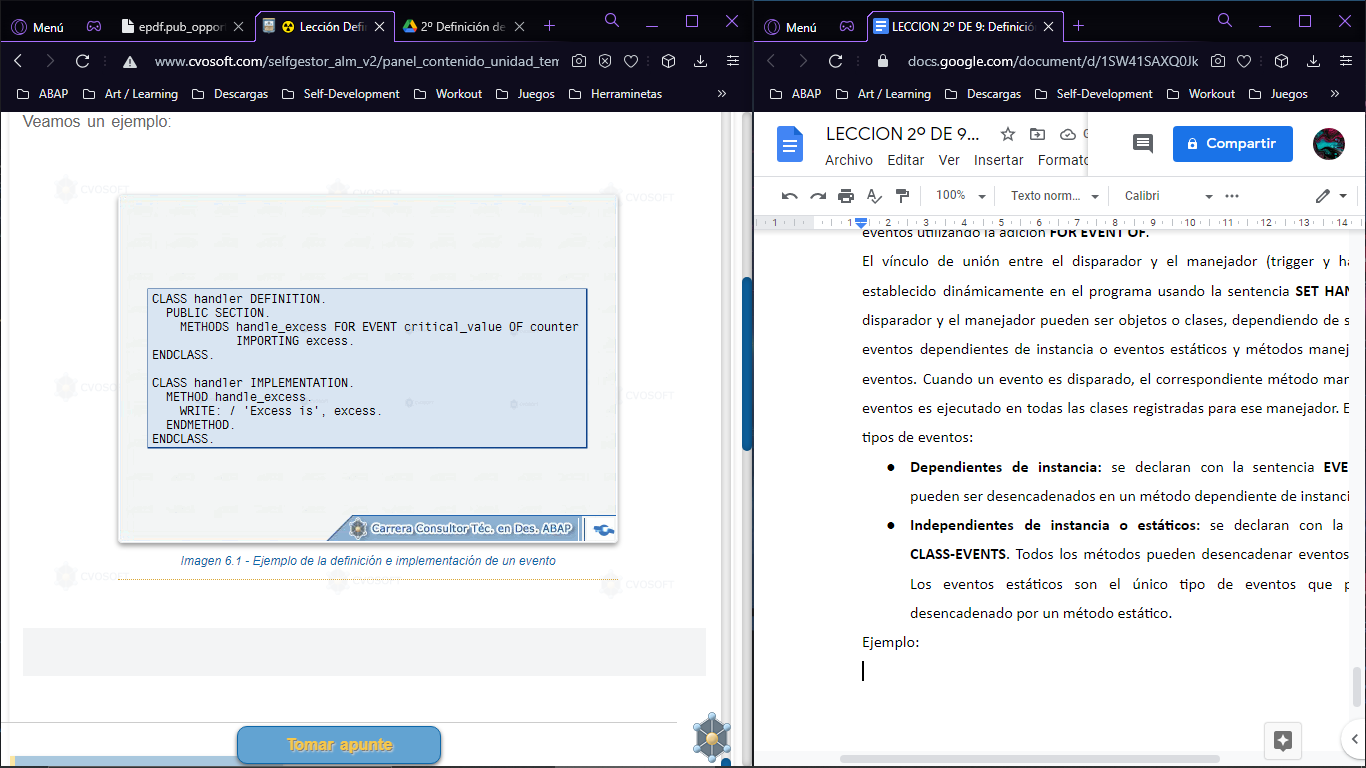
El manejador de eventos determina los eventos a los cuales tiene que reaccionar. No tiene por qué existir un método manejador de eventos registrado para cada evento. Los eventos de una clase pueden ser desencadenados en los métodos de la misma clase usando la sentencia **RAISE EVENT.**

Un método de la misma o de diferente clase, se declara como método manejador de eventos utilizando la adición **FOR EVENT OF**.

El vínculo de unión entre el disparador y el manejador (trigger y handler) es establecido dinámicamente en el programa usando la sentencia **SET HANDLER**. El disparador y el manejador pueden ser objetos o clases, dependiendo de si tenemos eventos dependientes de instancia o eventos estáticos y métodos manejadores de eventos. Cuando un evento es disparado, el correspondiente método manejador de eventos es ejecutado en todas las clases registradas para ese manejador. Existen dos tipos de eventos:

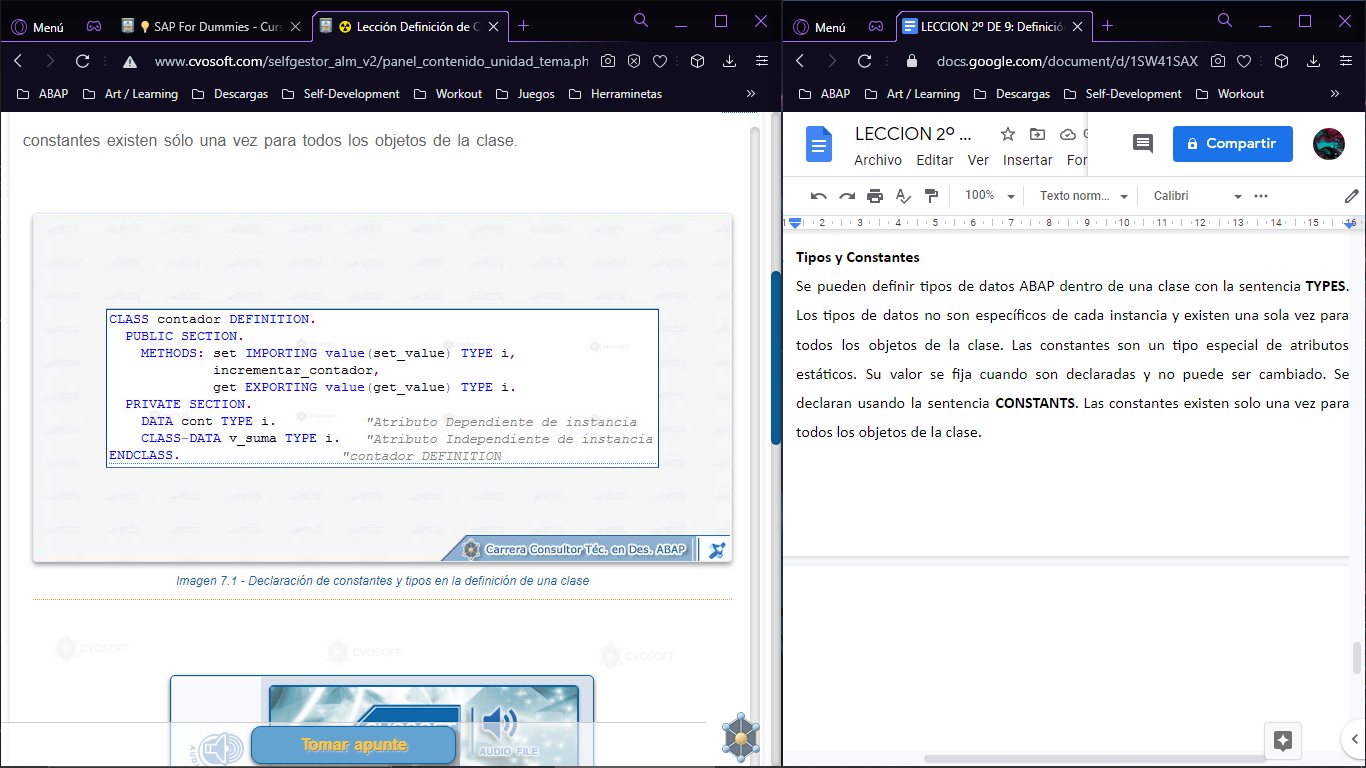
* **Dependientes de instancia:** se declaran con la sentencia **EVENTS**. Solo pueden ser desencadenados en un método dependiente de instancia.
* **Independientes de instancia o estáticos:** se declaran con la sentencia **CLASS-EVENTS**. Todos los métodos pueden desencadenar eventos estáticos. Los eventos estáticos son el único tipo de eventos que puede ser desencadenado por un método estático.

Ejemplo:



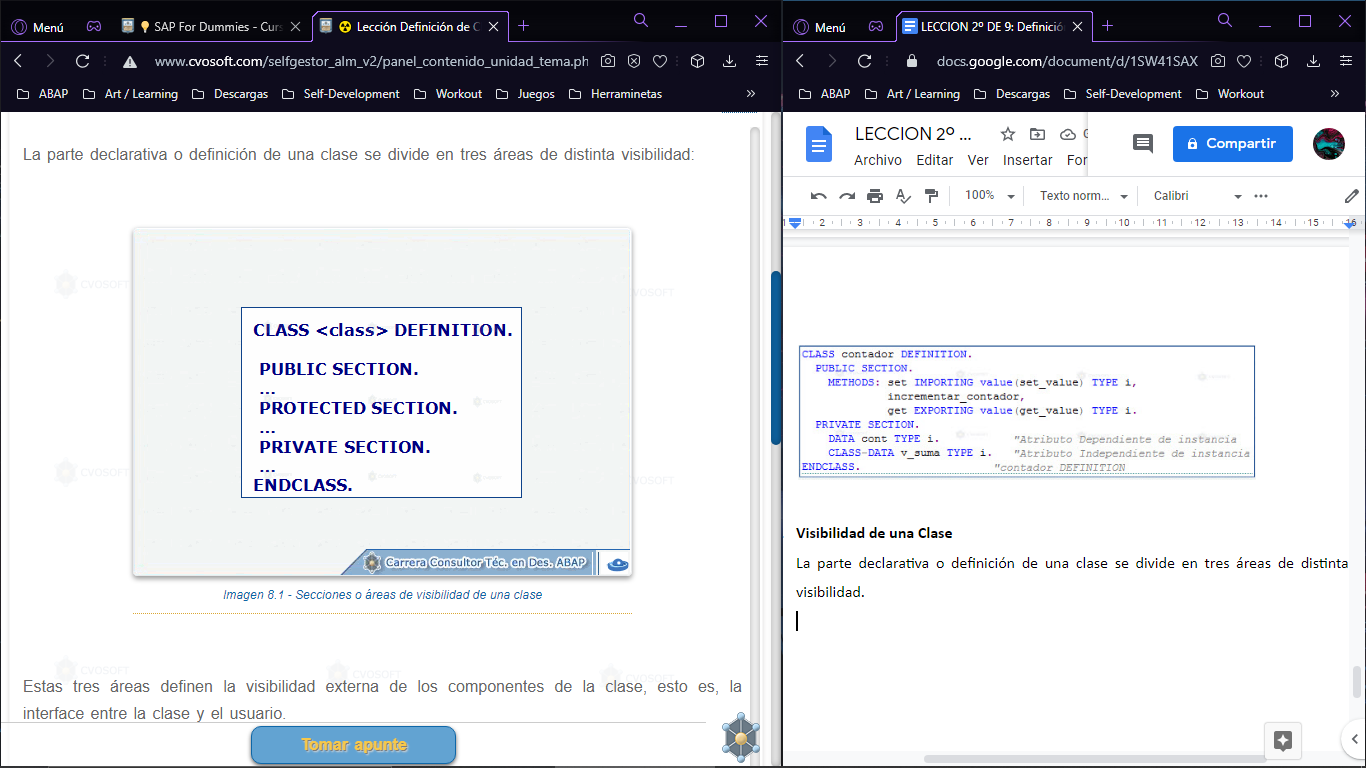
1. **Tipos y Constantes**

Se pueden definir tipos de datos ABAP dentro de una clase con la sentencia **TYPES**. Los tipos de datos no son específicos de cada instancia y existen una sola vez para todos los objetos de la clase. Las constantes son un tipo especial de atributos estáticos. Su valor se fija cuando son declaradas y no puede ser cambiado. Se declaran usando la sentencia **CONSTANTS**. Las constantes existen solo una vez para todos los objetos de la clase.



1. **Visibilidad de una Clase**

La parte declarativa o definición de una clase se divide en tres áreas de distinta visibilidad.



Estas tres áreas definen la visibilidad externa de los componentes de la clase, esto es, la interface entre la clase y el usuario.

Cada componente de una clase ha de ser asignado a una de estas tres secciones:

* **PUBLIC SECCIÓN (Parte Pública):** todos los componentes declarados en la sección pública son accesible para todos los usuarios de la clase y para todos los métodos de la clase y de cualquier clase que herede de ella. Los componentes públicos conforman la interface entre la clase y el usuario.
* **PROTECTED SECTION (Parte Protegida):** todos los componentes declarados en la sección protegida son accesible para todos los métodos de la clase y de las clases que heredan de ella. Los componentes protegidos conforman la interface entre una clase y todos sus subclases.
* **PRIVATE SECTION (Parte Privada):** los componentes declarados en la sección privada son solo visibles en los métodos de la misma clase. Los componentes privados no forman parte de la interface externa de la clase.